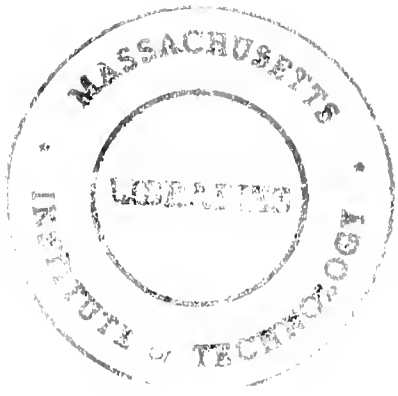

MASSACHUSETTS
INSTITUTE OF TECHNOLOGY

Identifying Controlling Features of Engineering Design Iteration

Robert P. Smith
Steven D. Eppinger

Revised September 1992
WP #3348-91-MS

Next revision: January 1993.
Please write to address below after that date for reprints.

Identifying Controlling Features of Engineering Design Iteration

Robert P. Smith
Steven D. Eppinger

Revised September 1992
WP #3348-91-MS

Acknowledgment

This research is funded by General Motors and by the Leaders for Manufacturing Program, a partnership involving eleven major U.S. manufacturing firms and M.I.T.'s schools of engineering and management. The authors are also grateful to Dan Whitney, Marcie Tyre, Karl Ulrich, and two anonymous reviewers from *Management Science* who have provided helpful and insightful comments on earlier versions of this paper.

Send correspondence to:
Prof. Steven D. Eppinger
M.I.T. Sloan School of Management
30 Wadsworth Street, E53-347
Cambridge, Mass. 02139

Abstract

Engineering design generally involves a very complex set of relationships among a large number of coupled elements. It is this complex coupling that leads to iteration among the various engineering tasks in a large project. The Design Structure Matrix (DSM) is useful in identifying where iteration is necessary. The Work Transformation Model developed in this paper is a powerful extension of the DSM method which can predict slow and rapid iteration within a project, and predict those features of the design problem which will require many iterations to reach a technical solution. This model is applied to an automotive brake system development process in order to illustrate the model's utility in describing the main features of an actual design process.

Introduction

The goal of this work is to develop a modeling framework which is useful for describing engineering design iteration. The framework is applied to brake system design to illustrate its utility in understanding the engineering design process.

Engineering design is the process whereby a technical solution is created to solve a given problem. There have been several attempts to give formal structure to the design process, such as those of Suh [1990], Pahl and Beitz [1988] and Alexander [1964]. This stream of research characterizes good design practice in general terms, but does not describe what makes some design problems more difficult than others. We intend to further the development of design process modeling by providing richness to the descriptions of design procedures and strategies which will enable a design organization to identify the difficult portions of their particular design problem. Strategies can then be developed to facilitate the effective execution of these difficult aspects.

The Design Structure Matrix (DSM) serves as the basis for our formal analysis and will be briefly reviewed in this section. (For a more detailed overview of the DSM method the reader is referred to Steward [1981] and Eppinger *et al.*

[1991].) The work herein extends the analytical method, and demonstrates the utility of our framework for the management of engineering projects.

The philosophy of the DSM method is that the design project is divided into individual tasks, and the relationships among these tasks can be analyzed to identify the underlying structure of the project. It has been suggested that studying the relationships between individual design tasks can improve the overall design process, and is a powerful way to analyze alternative design strategies [von Hippel 1990]. Earlier work developed a modeling formalism which shows how different aspects of a design problem are related [Alexander 1964]. Alexander describes a graphical technique where the functional needs of the technology are nodes, and interactions between the needs are arcs. His idea is to segment the graph into subsections which have relatively few interactions which cross boundaries. These graph segmentations give rise to technical subsystems which should separate the technical needs into independently solvable problems.

The DSM method is similar to Alexander's technique, but the nodes are now specific design tasks and the arcs are directed and indicate information flows between tasks. The nodes in the graph are arranged in a square matrix where each row and its corresponding column are identified with one of the tasks. Along each row, the marks indicate from which other tasks the given task requires input. Reading down each column indicates which other tasks receive its output. Diagonal elements do not convey any meaning at this point, since a task cannot depend upon its own completion. For example, in Figure 1 (based on a simplified view of camera body design), task **C** requires input from tasks **B**, **D**, **E** and **F**, task **B** requires input only from task **A**, and task **A** needs no input to begin.

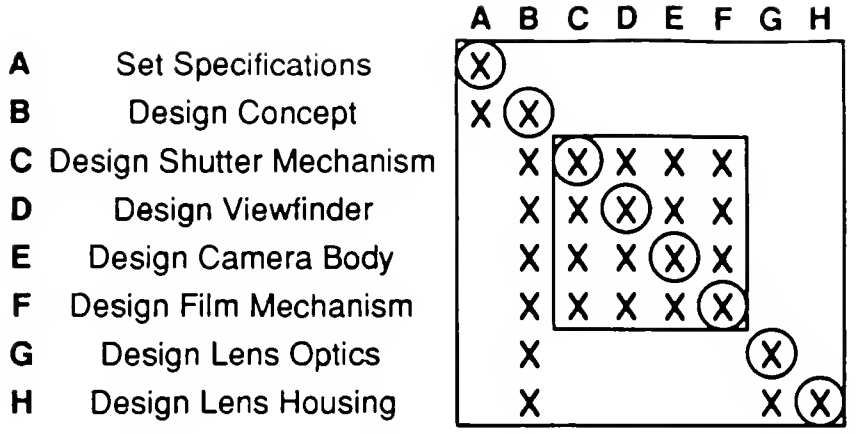


Figure 1. Sample Design Structure Matrix

The DSM can be used to identify orderings of tasks and to identify difficult aspects of the design process. Some or all of the elements of the matrix can be made sub-diagonal (such as those corresponding to tasks **A**, **B**, **G**, and **H** in Figure 1) by reordering the tasks of the matrix using a partitioning algorithm [Steward 1981, Gebala and Eppinger 1991]. An entirely sub-diagonal matrix indicates that there exists a sequence where all tasks can be completed with all input information available. Such a sequence may contain both tasks which must be done in series, or tasks which may be done in parallel. The information in a sub-diagonal design matrix is then similar to that expressed in a CPM (Critical Path Method) or PERT (Program Evaluation and Review Technique) chart.

More typically, due to the complexity in engineering design, the matrix cannot be reordered to have all matrix elements sub-diagonal (such as tasks **C-F** in Figure 1.) In these cases there is a cyclic flow of information in the design process and standard CPM/PERT techniques are not applicable because of the presence of such cycles. Likewise, a sequential progression of the design tasks is not possible. Tasks where neither a purely sequential nor a parallel ordering is feasible are coupled in such a way that some alternative process for resolving the design interactions (such as iteration or negotiation) must be used. For this

reason, iteration is a typical feature of engineering design projects [Hubka 1980]. The sub-matrix in Figure 1 depicts a design problem defined such that the tasks are sufficiently complex and interrelated so that iteration will be necessary to complete the tasks.

There is an established set of models which allow looping within a PERT modeling framework. This set of models is known as GERT, for General Evaluation and Review Technique. Direct analysis of any but a simple GERT network rapidly becomes unwieldy, so simulation is typically used to evaluate a project. (Taylor and Moore [1980] discuss the application of GERT to R&D projects.) It is the intention of this modeling effort to provide an analytically tractable model of the design iteration process, even for large projects. It is hoped that by preserving tractability it will be possible to observe the relationship between the structure of the problem and the development time of the project. Because GERT relies on simulation for large projects, it is difficult to discern this relationship.

For our purposes, we assume that the tasks and interrelationships of a design problem are known and unchangeable during the course of the project. This assumption is reasonable for a firm is working on a design project in an area in which they have a significant degree of familiarity (the example of brake system design at General Motors, which serves as the basis for the application described in this paper, fits this category). The assumption is less true for a completely new or rapidly evolving technology.

There is evidence that some companies who are faced with the same design problem choose differing design strategies, which implies a different underlying design matrix. For example, to what extent they choose to work on tasks in series or parallel affects development time significantly [Clark and Fujimoto 1991].

Development time is an important measure in engineering design management. We believe that complex iteration is a major source of extended development time. While the Design Structure Method is a useful tool to identify the coupled blocks in which the complex iteration occurs, this work is intended to characterize how such iteration occurs.

If we include task durations in the DSM, we can use this description to estimate the total duration of the project. Series tasks can be evaluated by summing their individual times, and parallel tasks can be evaluated by finding the maximum of those task times. For the project characterized by the DSM in Figure 1, if the task time are a, b, c, ... , h, the time of the camera design project would be

$$a + b + \max\{ f(c,d,e,f) , g+h \}$$

where f(·) is a function, undefined as yet, corresponding to the development time for the coupled block.

The model presented in this paper illustrates how iteration time can be evaluated for such a coupled block of tasks, and shows that the critical features controlling the iteration can be identified. Each critical feature is a group of parameters of the design solution which are strongly dependent on each other; they may require many iterations to converge, as a set, to conform to design constraints. We illustrate these concepts using a brake system design example. The critical features in brake system design are important determinants of product quality, and we believe that critical features which are strongly related to both time and quality are typical of engineering design.

Our Approach

We believe that it is possible to lessen development time by analyzing and restructuring the design process. We have developed extensions to the DSM

framework which have allowed us to suggest ways that the design process can be restructured. A previous interpretation of the quantitative DSM developed a probabilistic model of engineering design which predicts development time for a sequential design iteration process [Smith and Eppinger 1991], but that model has proven difficult to apply to actual design projects. The model presented in this paper is a different interpretation of quantitative information in the DSM, known as the Work Transformation Matrix (WTM), and is described below.

Our field work is based on extended exposure to the brake system design engineering organization the brake system design division of General Motors. Our observations include informal discussions with systems and component engineers, internal documentation, and interviews with engineers and their managers. We have found the brake system to provide a good subject for modeling of the design process because of the nature of the design problem. Brake system design is stable in that the technology and the market are mature and the form of the base product is not undergoing radical change. The brake system design engineers have considerable experience with brake system design. These factors suggest that the data contained within the brake system DSM is not changing rapidly, and the knowledge which is represented within the DSM is well developed.

Ours is a descriptive model, not an optimization model. The description developed below can be used by the design manager to analyze the design problem, and to estimate how long the design process will take, and what aspects of the design problem contribute to iteration time.

The novelty in this model is in the application of matrix mathematics to analyze development time of an iterative design process. The model relies on standard linear algebra results. The interpretation of the relationship between the matrix mathematics and development time is novel.

Design Iteration Model Development

For the purposes of our analysis, we assume that each task creates a deterministic amount of rework for other tasks. Rework is the work which is necessary because the task originally was attempted with imperfect information (assumptions). The rework adapts the original solution to account for the modified information. Rework is measured in percent of the time that it takes to complete the task in the original iteration.

We use a transformed version of a fully coupled Design Structure Matrix which we call the *Work Transformation Matrix* (WTM). There are two types of information in a Work Transformation Matrix. The off-diagonal elements represent strength-of-dependence measures (defined in next section). (See Figure 2a.) The diagonal elements in the WTM represent the time that it takes to complete each task during the first iteration stage. (See Figure 2b.) It is assumed that there will be multiple iteration stages, and that the time for each stage is a function of the amount of time spent working in the previous stage. We wish to find the sum of the times of all stages.

(a) Strength of Dependence Measures **(b) Task Times**

Figure 2. Work Transformation Matrix

The derivation below is divided into three sections. In the first section we describe the assumptions underlying the model. In the second section we describe why the eigenvalues and eigenvectors of the WTM are relevant to our analysis of development time. In the third section we describe how the eigenstructure of the matrix is interpreted. Following the derivation of the model,

we illustrate the analytical process using a simple example. Finally, we present and analyze the WTM which describes brake system design.

A: Work Transformation Model Assumptions

The assumptions in this model are:

- All tasks are done in every stage – fully parallel iteration
- Rework is created based on a linear rule – as a % of work done in previous iteration stage
- The parameters in the matrix describing work transformation behavior do not vary with time

These assumptions allow us to use a linear algebraic analytical method on the WTM.

To develop the model, we first introduce the concept of the work vector u_t. This is an n-vector, where n is the number of design tasks to be completed. Each element of the work vector contains the amount of work to be done on each task after iteration stage t. The initial work vector u_0 is a vector of ones, which indicates that all of the work remains to be completed on every task at the beginning of the iteration process.

During each iteration stage all work is completed on all of the design tasks. (For a relaxation of this assumption, where a fraction of the work is completed in every stage see Appendix 4.A.) However, work on a task will cause some rework to be created for all other tasks which are dependent on that task for information. We determine which tasks those are from the design structure matrix. Every iteration stage produces a change in the work vector according to:

$$u_{t+1} = Au_t$$

where each of the entries a_{ij} in A implies that doing one unit of work on design task j creates a_{ij} units of rework for design task i. The matrix A is then the

strength of dependencies portion of the WTM (Figure 2a). The diagonal entries are set to zero. The work vector u_t can be also be expressed by:

$$u_t = A^t u_0$$

The sum of each of the work vectors is the total work vector U, the total number of times that each of the tasks is attempted during the total of T iteration stages of design process:

$$U = \sum_{t=0}^{T} u_t$$

or:

$$U = \sum_{t=0}^{T} A^t u_0$$

which can be rewritten as:

$$U = \left(\sum_{t=0}^{T} A^t \right) u_0$$

The model output U is therefore in units of the original amount of work done on each task in the first iteration stage. (If element i in vector U is 1.6, then the design organization will have done 60% rework on task i in subsequent stages.) For a time-based interpretation of the matrix A see Appendix 4.B. For now, we scale U by the task durations to obtain units of task times. If W is a matrix which contains the task times along its diagonal (See Figure 2b), then WU is a vector which contains the amount of time (in engineer-hours) that each task will require during the first T iteration stages.

B: Eigenvalue Decomposition

If A has linearly independent eigenvectors (the eigenvector matrix S is invertible) then we can decompose A into:

$$A = S\Lambda S^{-1}$$

where Λ is a diagonal matrix of the eigenvalues of A, and S is the corresponding eigenvector matrix. (For S to be invertible it is sufficient, but not necessary, that none of the eigenvalues be repeated.) The powers of A can be found by:

$$A^t = S\Lambda^t S^{-1}$$

The total work vector U can therefore be expressed as:

$$U = S\left(\sum_{t=0}^{T}\Lambda^t\right)S^{-1}u_0$$

If the magnitude of the maximum eigenvalue is less than one, then the design process will converge (i.e. as T increases to infinity the total work vector U remains bounded.) An eigenvalue greater than one corresponds to a design process where doing one unit of work at some task during an iteration stage will create more than one unit of work for itself at some future stage. Such a system is unstable and the vector U will not converge, instead growing without bound as T increases. (It is a sufficient, but not necessary, condition for stability that the entries in every row sum to less than one.)

A design process which does not converge would be one where there is no technically feasible solution to the given specifications, or one where the designers are not willing to compromise to find the technical solution. This situation is not likely in the design environments we are modeling, that is, routine design where the designers are responsible for bringing out a new variation of a known, technically successful product. The remainder of the discussion on Work Transformation Matrices is limited to problems where a technical solution exists and can be found in finite time (i.e. eigenvalues are less than 1.)

C: Interpreting the Eigenstructure

The eigenvalues and eigenvectors of matrix A determine the rate and nature of the convergence of the design process. Much can be learned about what controls the iteration by looking at the eigenvalues and eigenvectors as opposed to looking at the sequence of work vectors.

A *design mode* is defined as a group of design tasks which are very closely related, and working on any one of them creates significant work, directly or indirectly, for each of the other tasks within the mode. We use the eigenvalues and eigenvectors of matrix A to identify the design modes.

The magnitude of each eigenvalue of A identifies the rate of convergence of each design mode. The eigenvector corresponding to each eigenvalue characterizes the relative contribution of each of the various tasks to the body of work which converges, as a group, at a given rate.[1]

By the Perron-Frobenius Theorem (a fundamental result of matrix theory) we know that the largest magnitude eigenvalue of a coupled non-negative matrix will be real and positive [Marcus and Minc 1964]. Also, the eigenvector associated with this eigenvalue will have positive elements.

The slowest design mode (largest eigenvalue) will therefore have an eigenvector which is strictly positive. This design mode gives us little problem with interpretation. Other design modes are, however, less obvious. Also by the Perron-Frobenius Theorem, there is only one eigenvector which is strictly positive. We must be able to interpret negative and complex numbers in the eigenvectors as well as negative and complex eigenvalues.

---

[1] The interpretation of the eigenvalues and eigenvectors for design problems is similar to the eigenstructure analysis used to examine the dynamic motion of a physical system. In the discrete time description of linear dynamic systems, each eigenvalue corresponds to a rate of convergence of one of the modes of the system (a natural frequency.) The eigenvectors identify the mode shapes of natural motion, quantifying the participation of each of the state variables in each mode [Ogata 1967].

Recalling that the total work vector U is calculated by:

$$U = S\left(\sum_{t=0}^{T}\Lambda^t\right)S^{-1}u_0$$

we will look at each of the elements in the above formula for U to see how the eigenstructure of matrix A can be used to interpret the design modes. If we take the limit as T approaches infinity we can use the formula:

$$\lim_{T\to\infty}\sum_{t=0}^{T}\Lambda^t = (I - \Lambda)^{-1}$$

If the maximum eigenvalue is not close to one, then the limit will be approached within relatively few iterations. For the remainder of this discussion the limit will be used, although the analysis can also be completed for finitely many iterations.

This limit is also a diagonal matrix, where each entry along the diagonal corresponds to one eigenvalue and has the form:

$$\frac{1}{1 - \lambda}$$

where λ is an eigenvalue.

In the next two subsections, both real and complex eigenvalues are discussed. In subsection 4.2.3.3 the interpretation of eigenvectors is considered.

Real Eigenvalues

The function:

$$\frac{1}{1 - \lambda}$$

is strictly increasing over (-1,1). The graph of this function is shown in Figure 3.

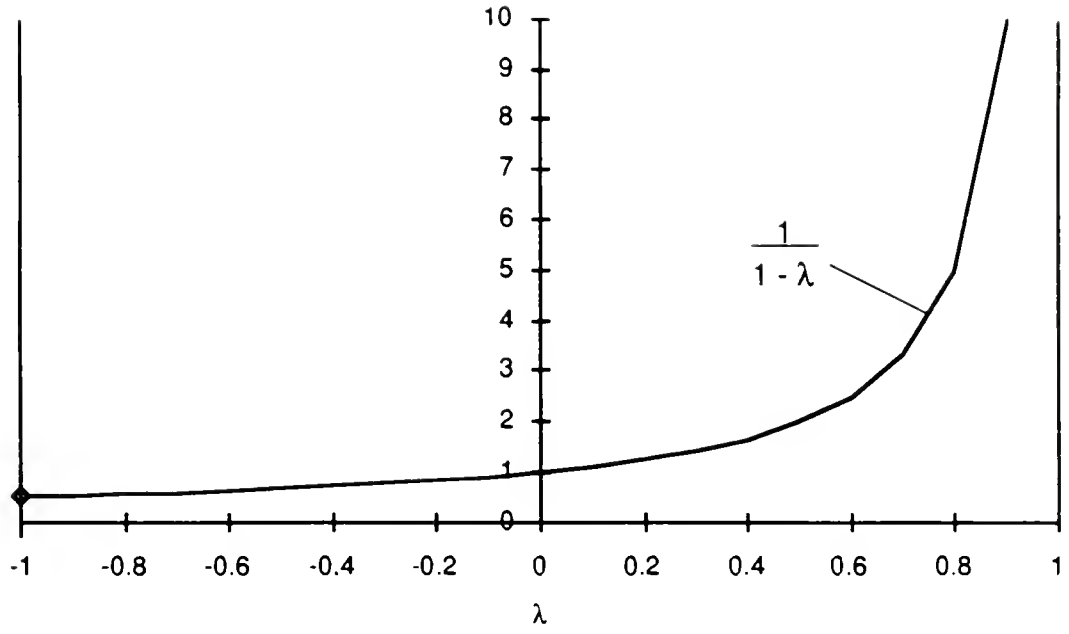


Figure 3. Graph of Magnitude vs. λ for Real Eigenvalues

We see that all positive eigenvalues have the greater contribution to the series sum than do negative eigenvalues. Therefore, as we consider which are the more important design modes, we restrict our attention among real eigenvalues to the positive eigenvalues.

Complex Eigenvalues

For complex eigenvalues we also wish to find the magnitude of the limit of the sum of the infinite series. For a complex eigenvalue $\alpha + \beta i$ the magnitude of the limit is:

$$\left| \frac{1}{1 - (\alpha + \beta i)} \right| = \frac{1}{\sqrt{(1 - \alpha)^2 + \beta^2}}$$

Or, alternatively:

$$\left| \frac{1}{1 - (\alpha + \beta i)} \right| = \frac{1}{\sqrt{1 - 2\alpha + \alpha^2 + \beta^2}}$$

Which, using the fact that:

$$\beta \neq 0$$

allows us to find an upper bound on the limit:

$$\left| \frac{1}{1 - (\alpha + \beta i)} \right| < \frac{1}{1 - \alpha}$$

Also, we can find a lower bound using the fact that:

$$\alpha^2 + \beta^2 < 1$$

to show:

$$\left| \frac{1}{1 - (\alpha + \beta i)} \right| > \frac{1}{\sqrt{2 - 2\alpha}}$$

The graph of the upper and lower bounds is shown in Figure 4.

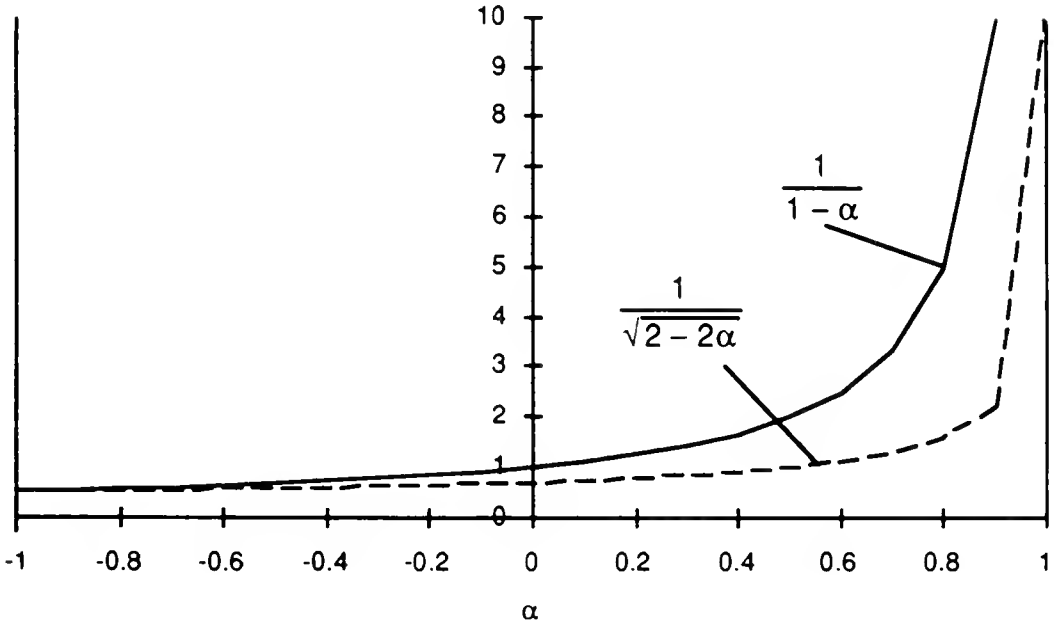


Figure 4. Graph of Bounds on Magnitude vs. α for Complex Eigenvalues

We see that the real part of complex eigenvalues gives bounds on the magnitude of the sum of the infinite series corresponding to that eigenvalue. We also see that complex eigenvalues with negative real part are not going to contribute significantly to the sum, and can therefore be ignored.

By the previous argument we need only consider those real eigenvalues which are positive. We therefore need to consider only those eigenvalues which have a positive real component, whether they are real or complex.

The Eigenvectors

This section discusses how the relative importance of each task within an eigenvectors is interpreted, given that we know the eigenvalue corresponding to that eigenvector. We want to be able to interpret the eigenvectors so that we can distinguish which of the tasks are important contributors to each design mode.

Again, consider the formula:

$$U = S\left(\sum_{t=0}^{T}\Lambda^{t}\right)S^{-1}u_0$$

We see that the final two terms in this formula:

$$S^{-1}u_0$$

give a weight for each eigenvector which is both a magnitude and a direction.

The eigenvector corresponding to real eigenvalues is real. Each weight for a real eigenvector is also real. Therefore, the direction is either positive or negative. The important quantities in a real eigenvector are therefore the large positive values if the weight is positive, and large negative values if the weight is negative.

Complex eigenvalues have complex eigenvectors and complex weights. Determining how the direction of the weight and the direction of the eigenvector interact is difficult. The best way to look at the interaction is to calculate the contribution of the mode to the total work vector U and see which the tasks give large contribution to the total work.

Positive eigenvalues correspond to non-oscillatory design modes. Negative and complex eigenvalues describe damped oscillations. Oscillatory design modes

indicate that the work is not decreasing for all of the tasks in the mode at the same rate, but that the work is shifting from task to task during iteration process.

The magnitude of the variability in the amount of work between separate work vectors is not as important as the total magnitude of work completed. The specifics of the variability would be useful if we were tracking the individual task work information. Instead we are looking at aggregate information, so the individual variability (as indicated by the non-positivity of the eigenvector or eigenvalue) is less important. As we interpret the modes of the design process we must therefore concentrate on those modes with large positive real eigenvalues, or imaginary eigenvalues with a large positive real part.

An illustration of the interpretation of the eigenvalues and eigenvectors is given in the next section, where an example problem is fully worked.

A Simple Example

As an illustration of the above discussion, let us consider the following 4x4 Work Transformation Matrix. This is a quantitative version of the coupled block (tasks **C-F**) in the camera design matrix as shown in Figure 1. The tasks in this matrix are, in order: Design Shutter Mechanism, Design Viewfinder, Design Camera Body, and Design Film Mechanism. The numbers can be interpreted as follows: if the shutter is completely redesigned, then 30% of the viewfinder design work must be redone (entry in row 2, column 1 is 0.3), and so forth.

$$A = \begin{bmatrix} 0 & 0.1 & 0.2 & 0.3 \\ 0.3 & 0 & 0.4 & 0.2 \\ 0.1 & 0.3 & 0 & 0.5 \\ 0.1 & 0.1 & 0.2 & 0 \end{bmatrix}$$

The eigenvalue (Λ) and eigenvector (S) matrices are:

$$\Lambda = \begin{bmatrix} 0.674 & & & \\ & -0.392 & & \\ & & -0.141+0.060i & \\ & & & -0.141-0.060i \end{bmatrix}$$

$$S = \begin{bmatrix} 0.410 & -0.067 & 0.657 & 0.657 \\ 0.624 & -0.613 & 0.060 - 0.570i & 0.060 + 0.570i \\ 0.580 & 0.758 & -0.395 + 0.073i & -0.395 - 0.073i \\ 0.326 & -0.213 & -0.065 + 0.274i & -0.065 - 0.274i \end{bmatrix}$$

The four eigenvectors are the columns in S. Each vector in S has as its eigenvalue the associated diagonal element of Λ. The eigenvectors are (arbitrarily) scaled to be unit vectors. By inspection of the eigenvectors, we learn that the most slowly converging design mode (the one with the largest magnitude eigenvalue) involves primarily the middle two tasks. When we compute the first few work vectors, we find that they support the above interpretation.

$$u_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad u_1 = \begin{bmatrix} 0.6 \\ 0.9 \\ 0.9 \\ 0.4 \end{bmatrix} \quad u_2 = \begin{bmatrix} 0.39 \\ 0.62 \\ 0.53 \\ 0.33 \end{bmatrix} \quad u_3 = \begin{bmatrix} 0.267 \\ 0.395 \\ 0.390 \\ 0.207 \end{bmatrix} \quad u_4 = \begin{bmatrix} 0.180 \\ 0.278 \\ 0.249 \\ 0.144 \end{bmatrix}$$

The work done on the first and last tasks is less than the work on the middle two tasks during all iteration stages. We see that the *dominant mode*, the eigenvector of the most slowly converging eigenvalue, dominates the shape of convergence of the work vectors.

We can see what is happening by looking in more detail at the intermediate calculations used to determine the total amount of work completed during the iteration process. The infinite sum of the eigenvector matrix shows that the one positive eigenvector will contribute significantly more work to the process than the negative and the complex modes:

$$(I - \Lambda)^{-1} = \begin{bmatrix} 3.065 & & & \\ & 0.718 & & \\ & & 0.874+0.046i & \\ & & & 0.874-0.046i \end{bmatrix}$$

The term used to see how each of the modes is represented in the original state vector is:

$$S^{-1}u_0 = \begin{bmatrix} 2.125 \\ -0.310 \\ 0.082 - 0.461i \\ 0.082 + 0.461i \end{bmatrix}$$

Multiplying this weighting vector by the sum of the eigenvalue matrix we find the total weight on the eigenvector matrix:

$$(I - \Lambda)^{-1}S^{-1}u_0 = \begin{bmatrix} 6.513 \\ -0.223 \\ 0.093 - 0.399i \\ 0.093 + 0.399i \end{bmatrix}$$

Note that the weight on the first eigenvector is significantly larger in magnitude than the other weights. Most of the work in this iteration process is described by the primary design mode.

We are now able to calculate the total work vector:

$$U = S(I - \Lambda)^{-1}S^{-1}u_0 = \begin{bmatrix} 2.807 \\ 3.755 \\ 3.595 \\ 2.375 \end{bmatrix}$$

There has been more work completed during the process by the middle two tasks, just as the preliminary analysis of the eigenvectors and eigenvalues indicated.

Brake System Design

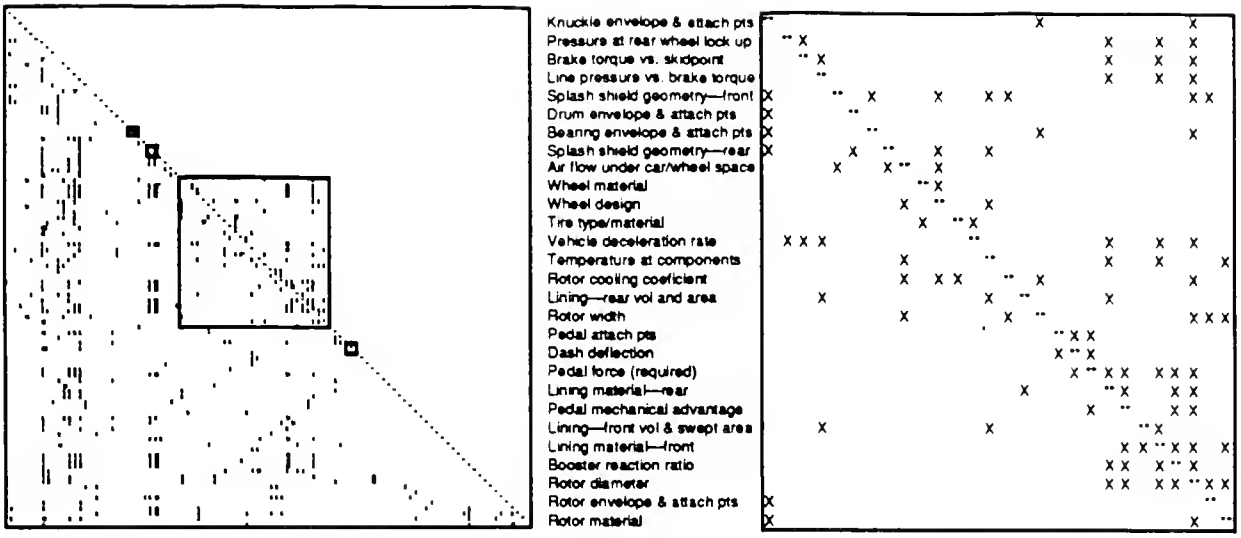
In order to verify the utility of the Work Transformation technique, we will demonstrate the analysis of an actual design process and show the insights gained. A design structure matrix for the brake system was reported previously [Black 1990, Black *et al.* 1990.] The work described here applied the Work Transformation Matrix method to the brake system design process. In preparing

this analysis, the first author spent several months doing field work at the brake system design facility of General Motors. That work included interviews with brake system engineers at several sites.

There are four questions which must be answered in constructing the Design Structure Matrix. We must first determine all of the various steps or tasks in the design process. Second, we must determine all of the information flows between the various tasks. Third, we must determine the relative importance of each of the information flows (quantifying the off-diagonal elements in the matrix). Fourth, we must estimate the time it takes to complete each task.

The brake system data presented in this paper includes data of the first three types, but does not include any explicit time data. Many of the observations about the controlling features of the design process can be made without having the time data available. In particular, we are able to identify the total number of iterations taken on each task.

The brake system DSM from Black *et al.* [1990] is shown in Figure 5a. The matrix demonstrates a problem which can be divided into a block of complex, coupled design parameters at the center of the matrix, preceded by and followed by a group of sequential and parallel parameters. The coupled block is expanded in Figure 5b. (We realize that Figure 5a is too small to see the details of the matrix; it is included to demonstrate the overall structure of the DSM, which includes over 100 design parameters.)



(a) Complete Matrix **(b) Expansion of Iterative Block**

Figure 5. Brake System Design Matrix

The interactions between design parameters are poorly understood, which leads to the coupling and the complexity of the design. If all interactions were well understood on a technical level, then the behavior could be described by predictive mathematical models (analytical or simulation). Iteration using such predictive models would be relatively fast. However, since there are many system level interactions which are not well enough understood to create a good predictive model, there are many lengthy iterations in the brake system design process. These iterations include costly and time-consuming experiments.

The Engineer's View

The customer wants an automobile with quiet, smooth brakes that do not require frequent service. To the engineer this means that the designed brake system should have little or no brake squeal or brake pulsation, and that the linings should have a long life. These problems are known respectively as noise, pulsation and wear. The generic causes of inability to meet these functional requirements are understood by engineers – stick-slip friction excites audible resonances (noise) in the rotor and other nearby structures, uneven rotor wear

leads to pulsation, and elevated lining temperature leads to rapid wear of the brake linings. More specific causes remain unknown. Detailed analysis of these problems continues, and some progress is being made. The sentiment among engineers is that none of these problems will be 'solved' in the near future. Specifically, brake systems cannot be designed so that no customers ever complain about these three problems. These problems are believed to be inherent consequences of using dry friction to stop a vehicle.

These three problems (noise, pulsation and wear) have been identified by designers and their managers as the 'controlling features' of the design/test/redesign iteration problem they experience. As shown below, the WTM analysis confirms that these are indeed controlling issues in design iteration, and details the specific contributing parameters for each. The match between designer perception and analytical identification lends credence to both approaches.

Using the Work Transformation Method to Identify Iteration Drivers

Using the analytical tools described previously, we can more rigorously identify the parameters within the large coupled block which compose the most interrelated sets of parameters. The original DSM analysis of the brake system identified parameters within the large block [Black 1990]. This work furthers the analysis by recognizing that some of the parameters exhibit stronger interdependence than others, and that tightly coupled parameters consequently require more iteration during the design process. The dominant design modes would be interpreted as 'controlling features' or 'design drivers', in that they require more engineering time during the design process and they are likely to be on the critical path of the design project.

To perform this analysis, we translate the binary DSM (Figure 5b) into a Work Transformation Matrix. In lieu of precise numerical values in the Work

Transformation Matrix for the brake system, the individual cells were estimated to be of either weak, medium, or strong dependence. (See Figure 6.) Each off-diagonal values is an estimate of the amount of work (as a percent of the amount of time that it took to complete the task during the original iteration) that the upstream task creates for the downstream task. The engineers in the design organization were asked to describe why each piece of information was necessary and the relative importance of each of the pieces of input information. Each of these information flows were classified by the authors as either a strong, medium or weak dependency. We then assigned numerical values to the dependencies which were described by the engineers. We have used the values 0.5, 0.25, 0.05 for strong, medium, and weak dependence, respectively. Our experience shows that the identification of the design drivers is robust against minor changes in the values entered in the matrix.[2]

---

2 This robustness can be demonstrated in two ways: (1) If we scale all of the values in A by a constant factor, the eigenvectors will be unchanged. The eigenvalues will simply scale, and our interpretation of the analysis will not change. (2) If we scale only one set of values (say strong dependence becomes 0.6 instead of 0.5), then there would be no significant changes to the resulting eigenstructure. More details on sensitivity of the eigenvectors to the weights are given in [Smith 1992].

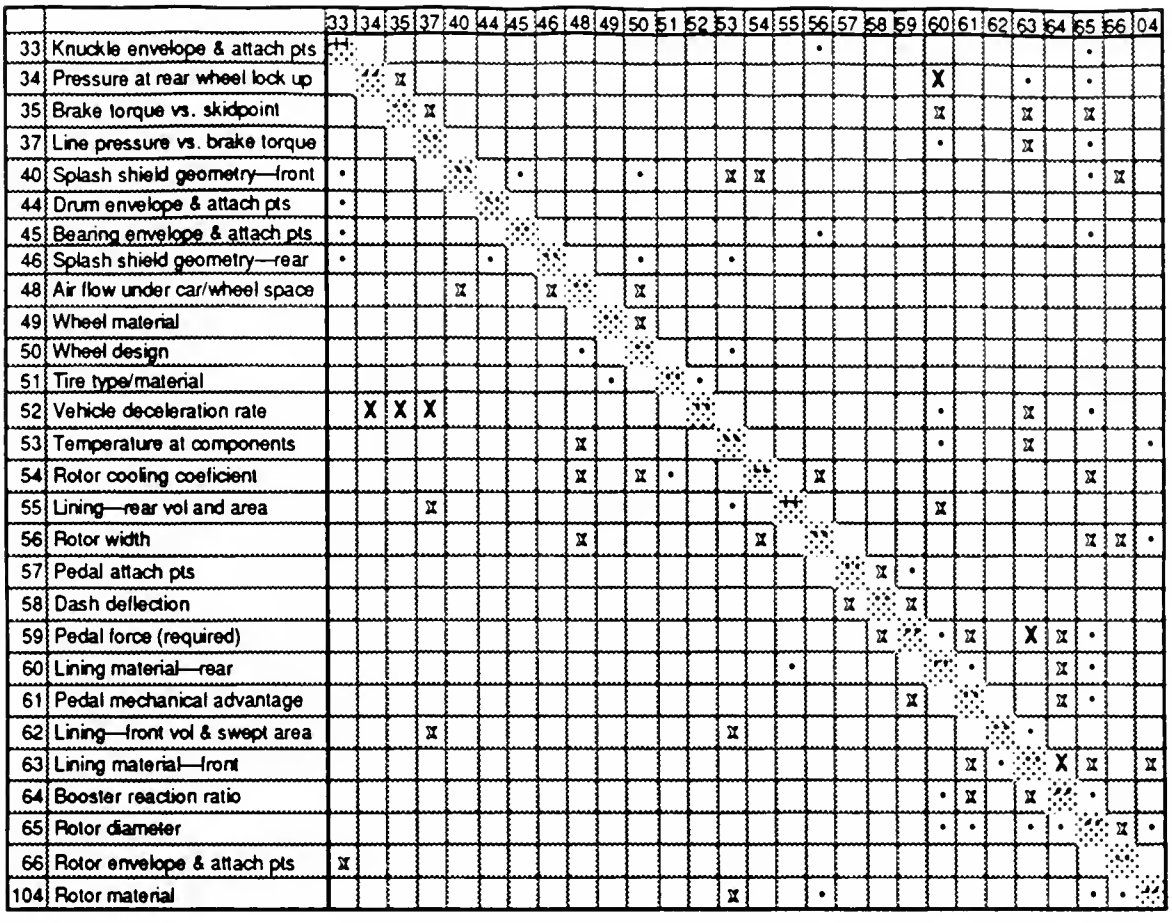


Figure 6. Coupled Block from Brake Matrix with Weighted Dependencies

The dominant design modes are represented by the eigenvectors of the work transformation matrix corresponding to the largest eigenvalues. Figure 7 shows the value of the quantity

$$\frac{1}{1 - \mathrm{Re}(\lambda)}$$

for each eigenvalue λ. This value is an upper bound on the magnitude of the number of times that each design mode will be repeated during the iteration process (as described earlier). Table 1 shows the values of the elements in the first two eigenvectors. (The larger magnitude elements are highlighted for emphasis and the eigenvectors are scaled to be unit vectors.)

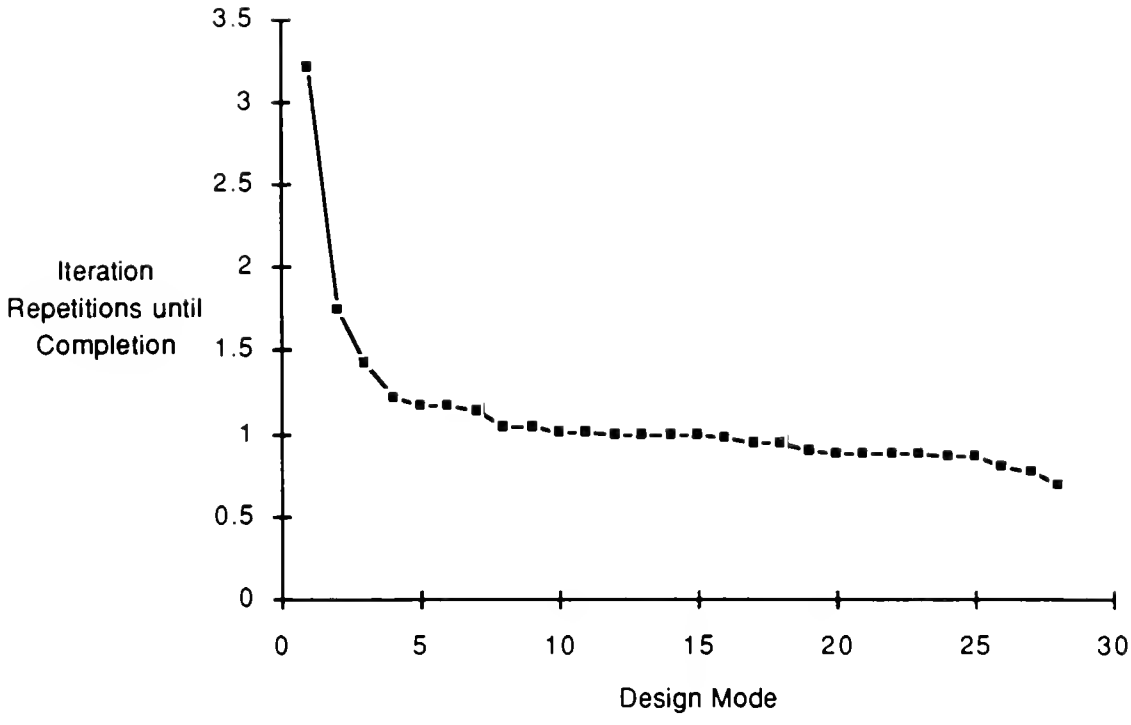


Figure 7. Brake System Eigenvalues

Task #	Parameter Name	First Design Mode	Second Design Mode
33	Knuckle envelope & attach pts	0.029	0.109
34	Pressure at rear wheel lock up	**0.305**	-0.065
35	Brake torque vs. skidpoint	**0.257**	-0.014
37	Line pressure vs. brake torque	0.125	-0.002
40	Splash shield geometry—front	0.099	**0.437**
44	Drum envelope & attach pts	0.004	0.023
45	Bearing envelope & attach pts	0.033	0.132
46	Splash shield geometry—rear	0.021	0.084
48	Air flow under car/wheel space	0.047	**0.326**
49	Wheel material	0.007	0.054
50	Wheel design	0.02	0.103
51	Tire type/material	0.078	-0.007
52	Vehicle deceleration rate	**0.58**	-0.087
53	Temperature at components	0.106	**0.168**
54	Rotor cooling coefficient	0.102	**0.464**
55	Lining—rear vol and area	0.096	0.006
56	Rotor width	0.101	**0.499**
57	Pedal attach pts	0.113	-0.118
58	Dash deflection	**0.174**	-0.154
59	Pedal force (required)	**0.414**	-0.178
60	Lining material—rear	0.123	-0.054
61	Pedal mechanical advantage	**0.219**	-0.122
62	Lining—front vol & swept area	0.114	0.088
63	Lining material—front	**0.28**	0.007
64	Booster reaction ratio	**0.198**	-0.066
65	Rotor diameter	0.121	0.024
66	Rotor envelope & attach pts	0.01	0.057
104	Rotor material	0.066	**0.209**
		Can we stop the car?	**Will the brake system overheat?**

Table 1. Brake System Eigenvectors

The first design mode is primarily composed of (52) Vehicle Deceleration Rate and (59) Pedal Force Required, with lesser input from (34) Pressure at Rear Wheel Lockup, (35) Brake Torque vs. Skidpoint, (58) Dash Deflection, (61) Pedal Mechanical Advantage, (63) Front Lining Material and (64) Booster Reaction Ratio.

What this design mode shows is that the group of design parameters which requires the greatest number of design iterations before convergence on the final acceptable design is the stopping distance problem, represented by the first column in Table 1. Solving this problem assures that the brake system is going to stop the car without creating uncontrollable skidding. A performance simulation for this problem has been developed, and it is a good predictor of actual performance. These iterations can therefore occur quickly. Because of this analytical tool, the large number of iterations on the first design mode no longer strongly affects the total time of the development process. This model nevertheless confirms that the stopping performance problem is the fundamental controlling feature which affects design iteration.

The second design mode is composed of primarily (40) Splash Shield Geometry, (48) Airflow under Wheel Space, (54) Rotor Cooling Coefficient and (56) Rotor Width, with lesser input from (53) Temperature at Components and (104) Rotor Material. All of these factors are technical parameters corresponding to overheating and cooling. This second design mode, which is composed of the cooling coefficient/rotor material problem, is related to the problems of lining life, noise generation, and pulsation problems. For these 'thermal' problems, there are few analytical or simulation tools available to the designer. Many iterations are required to converge upon a design solution, but there is no guarantee that those iterations can be rapid. Field or laboratory testing must be used to

eventually converge on a solution which meets the criteria at a relatively high cost in time.

The design modes analysis has been able to identify the dominant controlling features correctly. This success is made evident by the engineers' *a priori* prediction that noise, pulsation and wear would be found to be the fundamental design issues. We not only confirmed this, but also described these problems more precisely and showed how coupled these issues are.

Discussion and Conclusion

The goal of developing this type of model of engineering design is to be able to provide engineering managers information which will help to shorten the design cycle. Knowledge of the critical sets of interrelated tasks which lead to iteration enables a manger to concentrate resources on these tasks so that the iterations can occur as rapidly as possible.

The example of brake system design given above showed that the Work Transformation Model is able to match the observed behavior concerning which design tasks are responsible for the bulk of the iteration. The model is able to identify the features in the brake system design problem which control the total amount of time taken in the iteration process. The question remains open whether it is possible to identify the matrix data for a problem with which the design organization has less familiarity.

Applying the model to a new problem would provide new knowledge to the organization. The organization would be able to identify the critical issues prior to beginning the design process. There is the problem of whether or not the information required to construct the Work Transformation Model can be generated reliably on a new problem. We hypothesize that there is an important class of problems which are sufficiently well understood such that the engineers

involved can identify the tasks and the information dependencies (the information necessary to construct the matrix), without being able to identify the controlling features of the overall problem. This is the class of problems for which this type of analysis is relevant and useful.

This paper has developed the Work Transformation Model, which is an analytical extension to the Design Structure Matrix, and applied it to an actual design process. The analysis demonstrates the utility of eigenvalues and eigenvectors of the Work Transformation Matrix in interpreting the amount of work which must be completed during the design iteration process. The eigenvectors can be used to identify the 'controlling features', those elements of a coupled design problem which require the greatest number of iterations to reach a technical solution.

The Work Transformation Matrix can serve as a useful diagnostic tool in analyzing coupled design problems. We believe that this analytical method can lead to improvements in design processes by focusing attention on the slowly converging design iteration modes. For the brake system design process we suggest that improved simulation of the thermal and vibrational aspects of the design problem may accelerate solution development. In general, the identification of the controlling features provides a crucial piece of information which enables a manager to allocate resources in order to lessen development time.

References

Alexander, Christopher, *Notes on the Synthesis of Form*, Harvard University Press, Cambridge, 1964.

Black, Thomas A., "A Systems Design Methodology Applied to Automotive Brake System Design," Master's Thesis, M.I.T. Departments of Management and Mechanical Engineering, May 1990.

Black, Thomas A., Charles H. Fine and Emanuel Sachs, "A Method for Systems Design Using Precedence Relationships: An Application to Automotive Brake Systems," Working Paper, Leaders for Manufacturing Program, Massachusetts Institute of Technology, May 1990.

Clark, Kim B., and Takahiro Fujimoto, *Product Development Performance: Strategy, Organization, and Management in the World Auto Industry*, Harvard Business School Press, Boston, 1991.

Eppinger, Steven D., Daniel E. Whitney, Robert P. Smith and David A. Gebala, "Organizing the Tasks in Complex Design Projects," Sloan School of Management Working Paper 3083-89-MS, 1991.

Gebala, David A., and Steven D. Eppinger, "Methods for Analyzing Design Procedures," ASME Design Theory and Methodology Conference, Miami, 1991.

Hubka, Vladimir, *Principles of Engineering Design*, Butterworth Scientific, London, 1980.

Marcus, Marvin, and Henryk Minc, *A Survey of Matrix Theory and Matrix Inequalities*, Allyn and Bacon, Boston, 1964.

Ogata, Katsuhiko, *State Space Analysis of Control Systems*, Prentice Hall, Englewood Cliffs, N.J., 1967.

Pahl, Gerhard, and Wolfgang Beitz, *Engineering Design: A Systematic Approach*, Springer, New York, 1988.

Smith, Robert P., and Steven D. Eppinger, "A Model for Estimating Development Time of a Sequential Engineering Design Process," Sloan School of Management Working Paper 3160-90-MS, 1991.

Smith, Robert P., "Development and Verification of Engineering Design Iteration Models," Ph.D. Thesis, M.I.T. Sloan School of Management, August 1992.

Steward, Donald V., "The Design Structure System: A Method for Managing the Design of Complex Systems," *IEEE Transactions on Engineering Management*, Vol. EM-28, No. 3, pp. 71-74, 1981.

Suh, Nam P., *The Principles of Design*, Oxford University Press, New York, 1990.

Taylor, Bernard W., and Laurence J. Moore, "R&D Project Planning with Q-GERT Network Modeling," *Management Science*, Vol. 26, No. 1, pp. 44-59, 1980.

von Hippel, Eric, "Task Partitioning: An Innovation Process Variable," *Research Policy*, Vol. 19, pp. 407-418, 1990.

Appendix A

In these appendices are two extensions to the Work Transformation Model. It is shown here that the two extensions add generality to the original model, but are only slight modifications. The primary insight obtained from the analysis is that the eigenvalues and eigenvectors of A are still the most important analytical features, even with a more general model.

In the original model all of the work is executed during every iteration stage. We term this a control rule, since this is a work load policy. We can generalize the control rule. Instead of doing all of the work in every stage, we do a proportion ρ of all work on every task in each stage. The work which is not attempted during the current stage remains to be completed in future stages. Work which is attempted creates work for other tasks as in the original model. The new control rule becomes

$$u_{t+1} = [(1-\rho)I + \rho A]u_t \qquad 0<\rho\leq 1$$

We define a modified work transformation matrix A^* such that

$$A^* = [(1-\rho)I + \rho A]$$

We can find the eigenvectors and eigenvalues of the matrix A^*:

$$A^* = \left[(1-\rho)I + \rho S\Lambda S^{-1}\right]$$

$$A^* = S[(1-\rho)I + \rho\Lambda]S^{-1}$$

The matrix $[(1-\rho)I + \rho\Lambda]$ must be the eigenvalue matrix of A^* since it is diagonal. It is seen that the eigenvector matrix S of A^* is the same as that of A. The eigenvalues of A^* are a convex combination of Λ and I. Since the eigenvalues have been increased, the convergence has been slowed (which is to be expected since we

are only doing a proportion of the work in each stage.) The shape of the convergence remains unchanged.

Appendix B

The second extension treats time in a more explicit manner. It is shown below that this is, in fact, identical to the original way in which time was considered. The basis for the new formulation uses the vector $u^{\dagger}$ as a work time vector.

$$u^{\dagger}_{t+1} = A^{\dagger} u^{\dagger}_{t}$$

The initial work time vector is the initial work vector weighted by the time for each task:

$$u^{\dagger}_{0} = W u_{0}$$

where W is a diagonal matrix of the task times w_i.

Each element in the work time transformation matrix $A^{\dagger}$ is the amount of work time that one hour of task j creates for task i, or

$$a^{\dagger}_{ij} = \frac{w_i}{w_j} a_{ij}$$

The new work time transformation matrix is written compactly as

$$A^{\dagger} = WAW^{-1}$$

Repeating the analysis done for the original system, the total work time vector can be found by

$$U^{\dagger} = WS\left[\sum_{t=0}^{T} \Lambda^{t}\right] S^{-1} W^{-1} u^{\dagger}_{0}$$

Substituting for the initial work time vector

$$U^{\dagger} = WS\left[\sum_{t=0}^{T}\Lambda^{t}\right]S^{-1}W^{-1}Wu_0$$

$$U^{\dagger} = WS\left[\sum_{t=0}^{T}\Lambda^{t}\right]S^{-1}u_0$$

which reduces to

$$U^{\dagger} = WU$$

which is the expression originally given for weighting the total work vector by the task times.